

REMARKS

Claims 1-27 remain for consideration, and all claims are thought to be allowable over the cited art.

The Office Action does not establish that claims 1-27 are unpatentable under 35 USC §103(a) over "Enhanced Code Compression for Embedded RISC Processors" by Cooper et al. (hereinafter "Cooper"). The rejection is respectfully traversed because the Office Action fails to show that all the limitations are suggested by Cooper, fails to provide a proper motivation for modifying the teachings of Cooper, and fails to show that the modification could be made with a reasonable likelihood of success.

The invention is a method of optimizing computer program code where the computer program code includes a plurality of statements. The limitations include identifying a keyword statement; searching the program code for the keyword statement; determining if the keyword statement begins a repeating pattern of statements in the program code; and replacing the repeating pattern of statements with a program loop equivalent to the repeating pattern of statements. The Office Action does not establish that Cooper suggests these limitations.

For example, the cited section of Cooper does not suggest keyword statements or any searching for keyword statements in the program code. While the Office Action is correct in asserting that Cooper teaches finding sets of instructions that are repeated, Cooper does not rely on a keyword statement to determine whether instructions are repeated. Instead, Cooper builds a suffix tree in which each interior node identifies a repeated substring of the input. Before building the suffix tree, Cooper hashes each instruction and enters the instruction into a global table and creates a linear string-like representation of the program, where each character in the string corresponds to a particular instruction in the program. The suffix tree is constructed from the string representation of the program (p. 140, sections 2 and 2.1). Therefore, Cooper has

no apparent need for keyword statements since Cooper uses a suffix tree that is constructed from all the statements in the program without regard to any particular statement.

The limitations of replacing a repeating pattern of statements with an equivalent program loop is not obvious from Cooper because Cooper expressly replaces repeated patterns using either a procedural abstraction or cross-jumping. Neither the procedural abstraction nor the cross-jumping is suggestive of a program loop. Those skilled in the art will recognize that a program loop includes a control for repeatedly executing the loop until some condition is satisfied. In contrast, with Cooper's procedural abstraction a given code region is made into a procedure and other identical code regions are replaced with calls to the new procedure (page 141, section 3). With cross-jumping, identical regions that end with a jump to the same target are merged together by replacing one of the regions with a direct jump to the other region. Clearly, neither of these methods suggest use of a program loop since there is no control for limiting repetition of the loop.

The alleged motivation for modifying Cooper does not support *prima facie* obviousness. The alleged motivation states that "it would have been obvious ... when a repeated pattern of statements occurs a number of times in succession, replacing the repeated pattern with a jump instruction would have the effect of replacing the repeated pattern with an equivalent program loop. This alleged motivation is improper because, not only is it conclusory, but a simple jump statement does not by itself construct a program loop. Furthermore, no reasonable likelihood of successfully modifying Cooper has been shown.

As to claim 2, the further limitations include converting each data reference in a keyword statement to a data array reference. The Office Action does not establish that Cooper teaches these limitations at page 140. The cited section of Cooper teaches that each instruction with a unique combination of opcode, registers, and constants is entered into the global table. The instruction hashes to the entry in the table. In

contrast, the limitations state that the data reference in the keyword statement is converted to a data array reference. Thus, the invention includes changing the program code to include a data array reference in a keyword statement, whereas Cooper does not change the program code to include data array references.

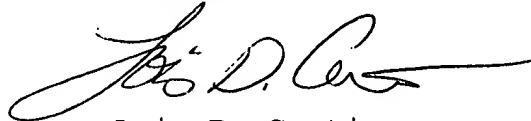
Claims 3 - 20 include limitations that further refine the limitations of claims 1 and 2. Therefore, the Office Action does not establish *prima facie* obviousness of claims 3 - 20. The Office Action does not establish a *prima facie* case of obviousness for claims 21-27 for at least the reasons set forth above because these claims include limitations similar to those in claims 1-20.

The rejection of claims 1-27 over Cooper should be withdrawn because the Office Action does not show all the limitations are suggested by the combination, does not provide a proper motivation for modifying Cooper, and does not show that Cooper could be modified with a reasonable likelihood of success.

#### CONCLUSION

Reconsideration and a notice of allowance are respectfully requested in view of the Remarks presented above. If the Examiner has any questions or concerns, a telephone call to the undersigned is invited.

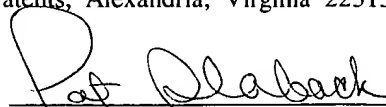
Respectfully submitted,



Lois D. Cartier  
Agent for Applicant  
Reg. No.: 40,941  
(720) 652-3733

I hereby certify that this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to: Mail Stop Amendment, Commissioner for Patents, Alexandria, Virginia 22313-1450, on August 12, 2004

Pat Slaback  
Name

  
Signature